

Processing system and method for transmitting data

The invention relates to a processing system.

The invention further relates to a method for transmitting data.

Systems on silicon show a continuous increase in complexity due to the ever increasing need for implementing new features and improvements of existing functions. This is enabled by the increasing density with which components can be integrated on an integrated circuit. At the same time the clock speed at which circuits are operated tends to increase too. The higher clock speed in combination with the increased density of components has reduced the area which can operate synchronously within the same clock domain. This has created the need for a modular approach. According to such an approach the processing system comprises a plurality of relatively independent, complex modules. In conventional processing systems the modules usually communicate to each other via a bus. As the number of modules increases however, this way of communication is no longer practical for the following reasons. On the one hand the large number of modules forms a too high bus load. On the other hand the bus forms a communication bottleneck as it enables only one device to send data to the bus. A communication network forms an effective way to overcome these disadvantages. The communication network comprises a plurality of partly connected nodes. Messages from a module are redirected by the nodes to one or more other nodes.

A message sent by a source functional unit may comprise a command or a packet of data. It is forwarded via one or more intermediate functional units until it arrives at the destination functional unit. The destination functional unit may on its turn send a message to the source functional unit. A functional unit can be any unit involved in a data stream for example a unit which performs operations on data, such as a CPU, a DSP or a VLIW, or a unit for storing data such as a memory, or a unit for transmitting data such as a router or an interface.

A split protocol is defined as a protocol where transactions are split in a request and a response. After a transmission of a request is completed from a source functional unit to the first intermediate functional in the communication path the source functional unit can proceed with a next transmission, instead of having to wait for a response

to that request from the destination functional unit. The destination functional unit will start a separate arbitration procedure if necessary to give a response. A split bus protocol is more efficient when a response generation at the slave takes time. Pipelining allows a master to have multiple outstanding requests (i.e., requests waiting for a response). All transactions within the same communication thread are ordered (responses are delivered in the same order as the requests for that responses where issued by a master). Transactions with different communication threads do not have any ordering constraints.

US 6,182,183 provides a link level protocol for exchanging the message between two subsequent functional units in the communication path from the source functional unit to the destination functional unit. According to the known protocol a master functional unit produces information, e.g. a command (Cmd), an address (Addr), or data (DataReq) and at the same time provides an identification of the thread (ReqThreadID) to which the information belongs. Likewise the slave functional unit may provide information (DataResp), and indicate the communication thread to which it belongs by an identification (RespThreadID).

It is a disadvantage of the known split, pipelined bus protocol that a data consuming functional unit only has limited control over the sequence in which it receives its data. If it has issued requests for data for several communication threads the data producing functional unit determines which of these threads is served first. This may have the consequence that the requested data arrives in an order which does not enable an optimal functioning of the data consuming functional unit.

It is a purpose of the invention to provide an improved link level communication protocol and an improved processing system using such a communication protocol. This purpose is achieved by a method according to the invention as claimed in claim 1, and by a processing system as claimed in claim 9. It is recognized by the inventors that it is in several circumstances advantageous if not the data producing functional unit, but the data consuming functional unit selects the communication thread for which information is exchanged. Hence, in a processing system and method according to the invention on the one hand a split transmission protocol in the communication path is applied. On the other hand for at least one pair of a data consuming and a data producing functional units in the

communication path which are communicating with each other the data consuming functional unit has direct control about the communication thread for which it receives data.

One example thereof is a data processing system, wherein the data consuming functional unit is a memory controller. A memory controller a.o. has the function to optimize the scheduling of storing/retrieving information from several communication threads, so that storage and retrieval can take place in a minimum amount of time. A memory controller according to the invention can very efficiently schedule the storage of information, as it can itself select the order of the threads for which it requests information. Alternatively it would be possible to provide the memory controller with large input buffers. This would enable it to receive large amounts of information from several threads and select information from the input buffers in a suitable sequence, but this would go at the cost of a reduced silicon area for other functions.

Another example where it is favorable that the data consuming functional unit can select the process thread is where the data consuming functional unit is a processor arranged for executing a plurality of tasks. Each task switch may require typically several hundreds to a thousand processing cycles. A task switch may typically occur if the processor has insufficient data available for a particular thread. In an embodiment of the invention the multitasking processor is capable of selecting a thread for which it requires data in order to continue with the same thread. In other words the processor schedules tasks taking into account their read data requirements, and indicates this to the data producing functional unit by means of the communication thread identifier.

In this way the frequency of task switches can be reduced

These and other aspects of the invention are described in more detail with reference to the drawing. Therein:

Figure 1 schematically shows a data processing system,

Figure 2 schematically shows a pair of a data producing functional unit and a data consuming functional unit in a communication path in the data processing system, according to the invention,

Figure 3 schematically shows a pair of a data producing functional unit and a data consuming functional unit in a second embodiment of the data processing system according to the invention,

Figure 4 schematically shows a pair of a data producing functional unit and a data consuming functional unit in a third embodiment of the data processing system according to the invention,

Figure 5 schematically shows a pair of a data producing functional unit and a data consuming functional unit in a fourth embodiment of the data processing system according to the invention.

Figure 1 schematically shows a data processing system, which comprises a network connecting a plurality of functional units. The processing system is arranged to transmit data and a communication thread identifier for said data according to a split protocol along a communication path (indicated by arrows) from a source functional unit SFU to a destination functional unit DFU via one or more intermediate functional units IFU1, ..., IFU5. By transmitting the data together with a communication thread identifier, multiple unrelated transactions, having mutually different communication thread identifiers can evolve independently.

Figure 2 schematically shows in more detail a pair of a data consuming functional unit CFU and a data producing functional unit PFU. By way of example the data consuming functional unit CFU and the data producing functional unit PFU respectively are the destination functional unit DFU and the fifth intermediate functional unit IFU5 in Figure 1.

The data consuming functional unit (CFU) and the data producing functional unit (PFU) in the communication path are arranged to directly communicate to each other by means of a handshake procedure. Therein the data consuming functional unit PFU issues a request for data REQ and indicates a communication thread identifier TID identifying for which communication thread it requests the data. It explicitly indicates validness of the TID with a signal TIDVAL. Alternatively validity of the request REQ and the communication thread identifier TID can be indicated by a particular value of one of these signals REQ, TID. If the data producing functional unit PFU has the data for the requested communication thread available it responds by indicating acceptance with the signal ACCEPTP and providing the requested data RESPDAT. In another embodiment the data producing functional unit accepts the request in a fixed number of clock cycles. A separate signal for indicating acceptance is then superfluous.

On its turn the data consuming functional unit CFU indicates acceptance of the data by a signal ACCEPTC. In another embodiment, where the data consuming functional unit always accepts the data within a fixed number of clock cycles a separate accept signal need not be generated by the data consuming functional unit.

5 Figure 3 shows an extended protocol. The protocol is initiated in the same way by the data consuming functional unit CFU, which provides an indication for a communication thread identifier TID and indicates that it is valid explicitly by a signal TIDVAL, or implicitly by reserving a particular value of the TID signal to indicate that it does not represent a valid communication thread. In this example the data consuming
10 functional unit CFU can only issue requests for reading data by providing a particular communication thread identifier. In case that it is desired that the data consuming functional unit CFU can also issue other commands, one or more additional request identification signals can be added to specify the command type.

Contrary to the embodiment shown in Figure 2 the data producing functional
15 unit PFU additionally can request the CFU to indicate an other communication thread identifier TID for which it wants to receive data. For that purpose the PFU provides a first and a second output signal ACCEPTP, READY which in a preferred embodiment have the following meaning:

READY	ACCEPTP	Meaning
0	*	The data consuming functional unit CFU has to continue indicating the communication thread identifier TID.
1	0	The second functional unit (CFU) is requested to indicate an other communication thread identifier.
1	1	The indicated communication thread identifier (TID) is accepted.

20 Issuing an acknowledge may take a number of cycles, in which time the READY signal is kept low (0). During that time the value of the signal ACCEPTP is not of importance (*). When the PFU is ready, READY is raised (1), and on the signal ACCEPTP indicates if the transaction has been accepted (1) or not (0). If a transaction is delayed (ACCEPTP = 0), it is possible to switch to another communication thread.

25 In the embodiment of Figure 3 there is one additional signal that encodes whether a link-level data exchange can proceed or whether it is delayed. With only one additional signal, there is only a proceed/delay feedback possible. This can be generalized to

encode more elaborate feedback on why a transaction cannot proceed on a communication thread (when there are multiple causes possible: e.g., empty/full buffers, a process not expecting data on a communication thread is running on a CPU, etc), or how long transactions can proceed on a communication thread (e.g., there is enough buffering/data to proceed with at least N transactions on a communication thread).

Still further embodiments of the invention are illustrated with reference to Figure 4 and 5. Therein the transmission method includes a further handshake procedure wherein information is exchanged from the data producing functioning unit (PFU) to the data consuming functional unit (CFU) to exchange communication thread information. The further handshake procedure is independent of the handshake procedure in which data is exchanged. In Figure 4 and 5 the signals TID, TIDVAL, RESPDAT, ACCEPTP and ACCEPTC have the meaning as described with reference to Figure 2. These signals form part of a first handshake procedure for exchanging data for a particular communication thread indicated by the data consuming functional unit CFU. The communication thread information provided by the data producing functional unit PFU can be used by a scheduler in the CFU to schedule better the sequence of communication threads for which it requests data. The information can for example be the amount of data available for that communication thread or the expected time interval whereafter new data will become available.

In the embodiment of Figure 4 the further handshake procedure includes an information request signal INFTID controlled by the data consuming functional unit CFU. This signal indicates a particular communication thread for which the data consuming functional unit indicates additional information. The data consuming functional unit CFU signals validity of the information request signal INFTID by a signal INFVAL. In a fixed number of clock cycles the data producing functional unit PFU provides the information TIDINFO relating to the specified communication thread. Instead of providing the information TIDINFO after a fixed number of clock cycles, it can be provided in a variable time interval if the data producing functional unit PFU provides a signal indicating the validity of this information. In the embodiment of Figure 4 the consumer functional unit CFU polls for information with the PFU independently of the transfer of data because that takes place in an independent handshake (TID, TIDVAL, RESPDAT, ACCEPTP, ACCEPTC).

Figure 5 shows an embodiment wherein the PFU controls the transfer of communication thread information in the further handshake procedure. The PFU indicates a communication thread about which it has information with signal INFTID and provides the information with the signal TIDINFO. Validity of these signals is indicated with the signal

INFVAL. The PFU may provide the information about all communication threads at each handshake, but independently of the transfer of data because that takes place in an independent handshake (TID, TIDVAL, RESPDAT, ACCEPTP, ACCEPTC). Alternatively it may provide that information for one communication thread at a time. In a preferred
5 embodiment only information is provided if the status of a communication thread has changed.

It is remarked that the scope of protection of the invention is not restricted to the embodiments described herein. It is noted that information, e.g. data for a communication thread, information about a communication thread, can be exchanged between the processing
10 units in several ways, e.g. serial, parallel or in a combination of ways.

Neither is the scope of protection of the invention restricted by the reference numerals in the claims. The word 'comprising' does not exclude other parts than those mentioned in a claim. The word 'a(n)' preceding an element does not exclude a plurality of those elements. Means forming part of the invention may both be implemented in the form of
15 dedicated hardware or in the form of a programmed general purpose processor. The invention resides in each new feature or combination of features.